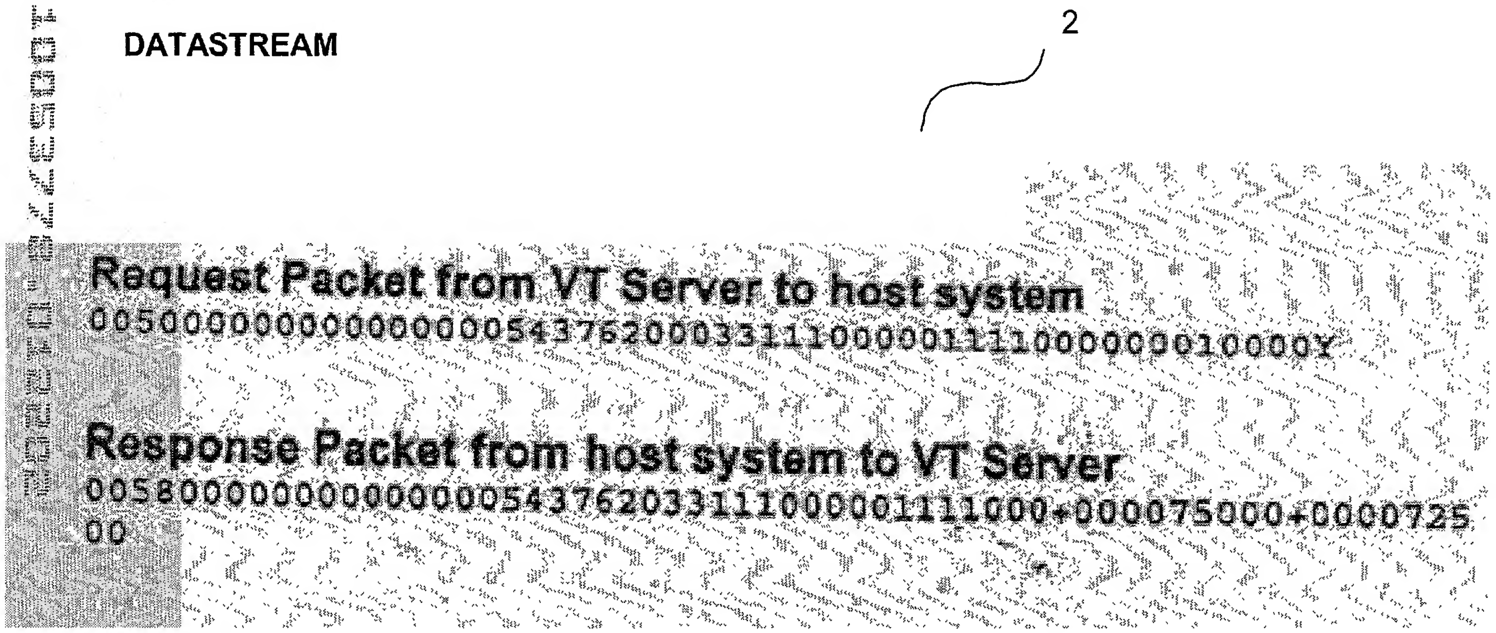
DATASTREAM



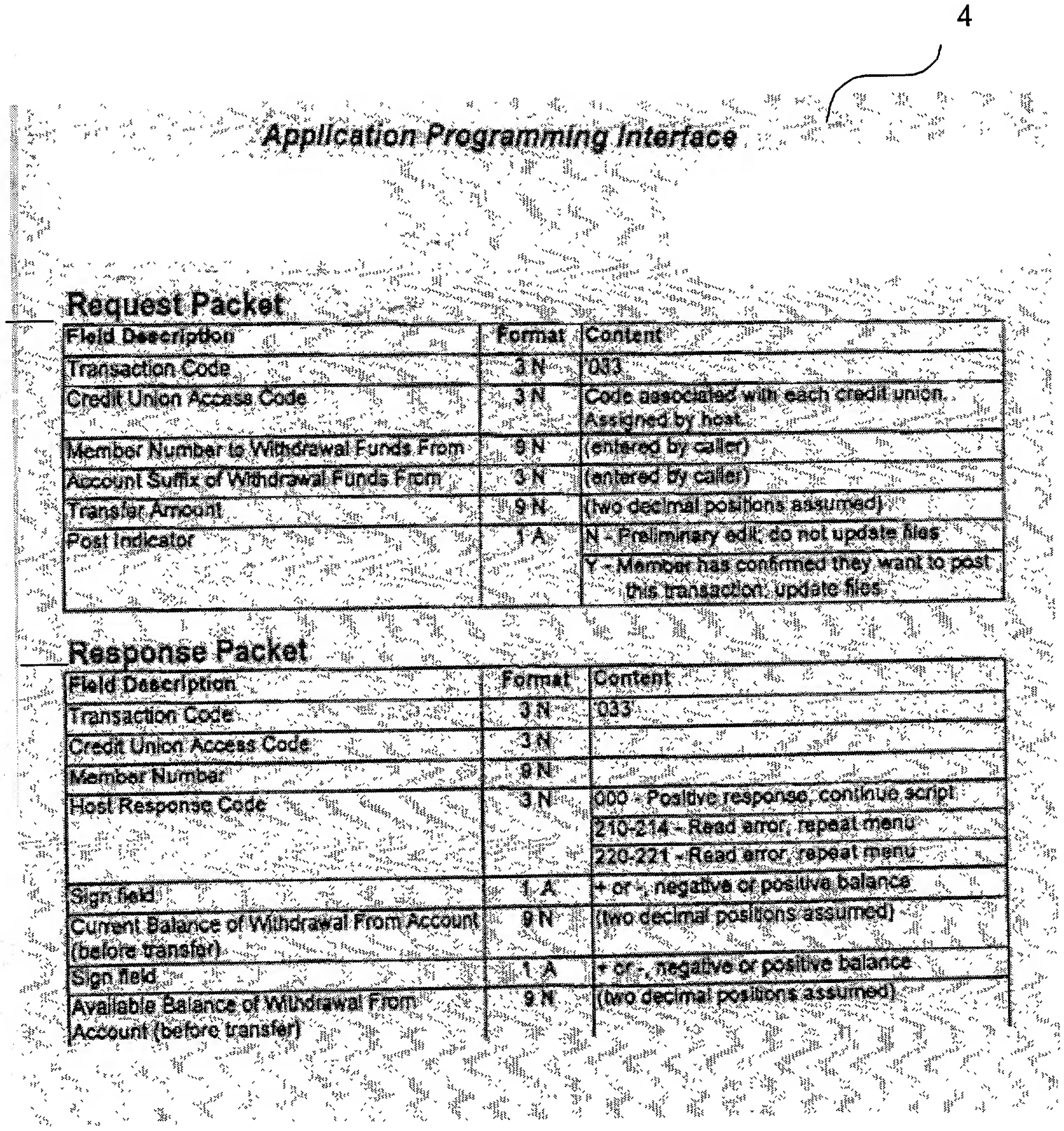
Request Packet from VT Server to host system

```
005000000000000000543762000331110000011110000000100000Y
```

Response Packet from host system to VT Server

```
005800000000000000543762033111000001111000+000075000+0000725
00
```

Fig.1



Application Programming Interface

Request Packet

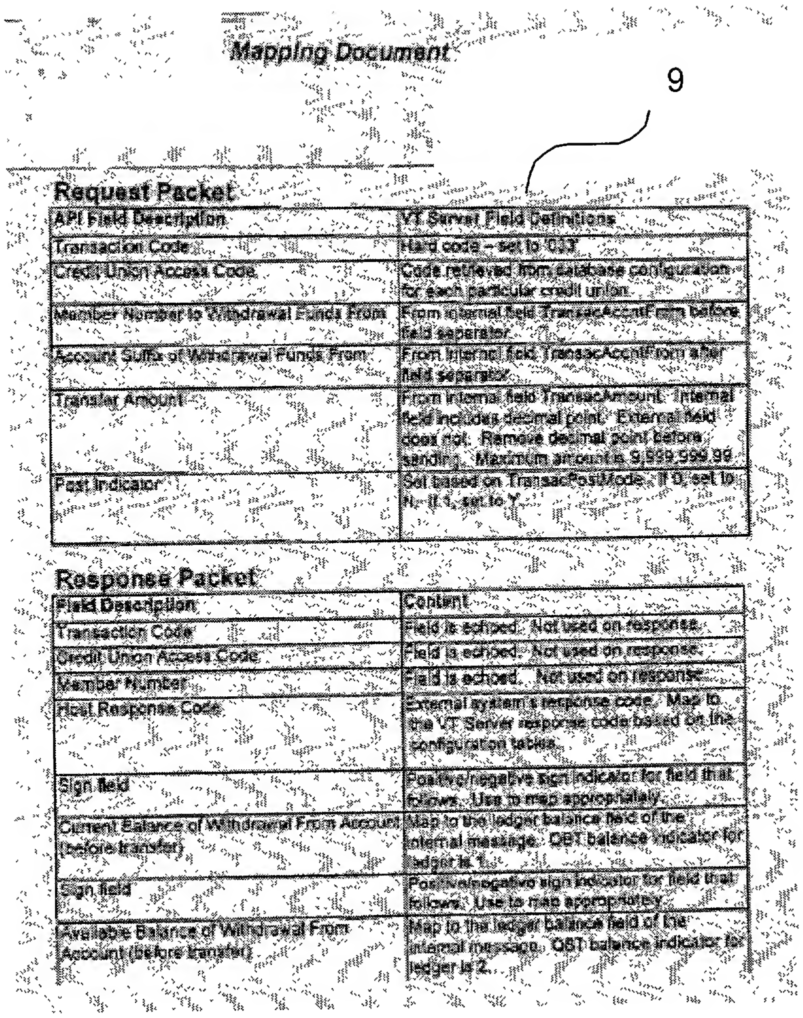
Field Description	Format	Content
Transaction Code	3 N	'033
Credit Union Access Code	3 N	Code associated with each credit union. Assigned by host.
Member Number to Withdrawal Funds From	9 N	(entered by caller)
Account Suffix of Withdrawal Funds From	3 N	(entered by caller)
Transfer Amount	9 N	(two decimal positions assumed)
Post Indicator	1 A	N - Preliminary edit, do not update files
		Y - Member has confirmed they want to post this transaction, update files

Response Packet

Field Description	Format	Content
Transaction Code	3 N	'033
Credit Union Access Code	3 N	
Member Number	9 N	
Host Response Code	3 N	000 - Positive response, continue script
		210-214 - Read error, repeat menu
		220-221 - Read error, repeat menu
Sign field	1 A	+ or -, negative or positive balance
Current Balance of Withdrawal From Account (before transfer)	9 N	(two decimal positions assumed)
Sign field	1 A	+ or -, negative or positive balance
Available Balance of Withdrawal From Account (before transfer)	9 N	(two decimal positions assumed)

Fig.2

10045376.032602



Mapping Document

9

Request Packet

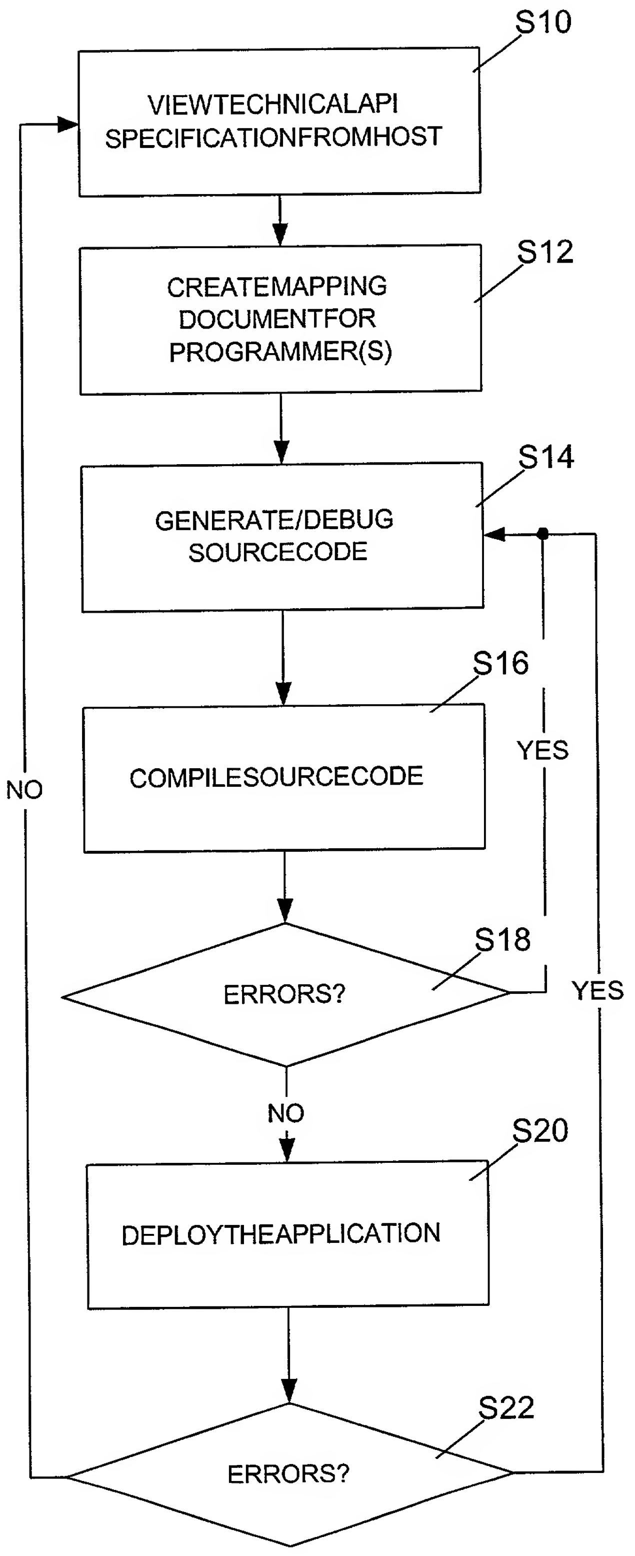
API Field Description	VT Server Field Definitions
Transaction Code	Hard code – set to '033'
Credit Union Access Code	Code retrieved from database configuration for each particular credit union.
Member Number to Withdrawal Funds From	From internal field TransacAccntFrom before field seperator.
Account Suffix of Withdrawal Funds From	From internal field TransacAccntFrom after field separator.
Transfer Amount	From internal field TransacAmount. Internal field includes decimal point. External field does not. Remove decimal point before sending. Maximum amount is 9,999,999.99
Post Indicator	Set based on TransacPostMode. If 0, set to N. If 1, set to Y.

Response Packet

Field Description	Content
Transaction Code	Field is echoed. Not used on response.
Credit Union Access Code	Field is echoed. Not used on response.
Member Number	Field is echoed. Not used on response.
Host Response Code	External system's response code. Map to the VT Server response code based on the configuration tables.
Sign field	Positive/negative sign indicator for field that follows. Use to map appropriately.
Current Balance of Withdrawal From Account (before transfer)	Map to the ledger balance field of the internal message. OBT balance indicator for ledger is 1.
Sign field	Positive/negative sign indicator for field that follows. Use to map appropriately.
Available Balance of Withdrawal From Account (before transfer)	Map to the ledger balance field of the internal message. OBT balance indicator for ledger is 2.

PRIORART

Fig.3


S10
VIEWTECHNICALAPI SPECIFICATIONFROMHOST
S12
CREATEMAPPING DOCUMENTFOR PROGRAMMER(S)
S14
GENERATE/DEBUG SOURCECODE
S16
COMPILESOURCECODE
YES
NO
S18
ERRORS?
YES
NO
S20
DEPLOYTHEAPPLICATION
S22
ERRORS?

PRIORART

Fig.4

20225140 " 8XZ55DGT

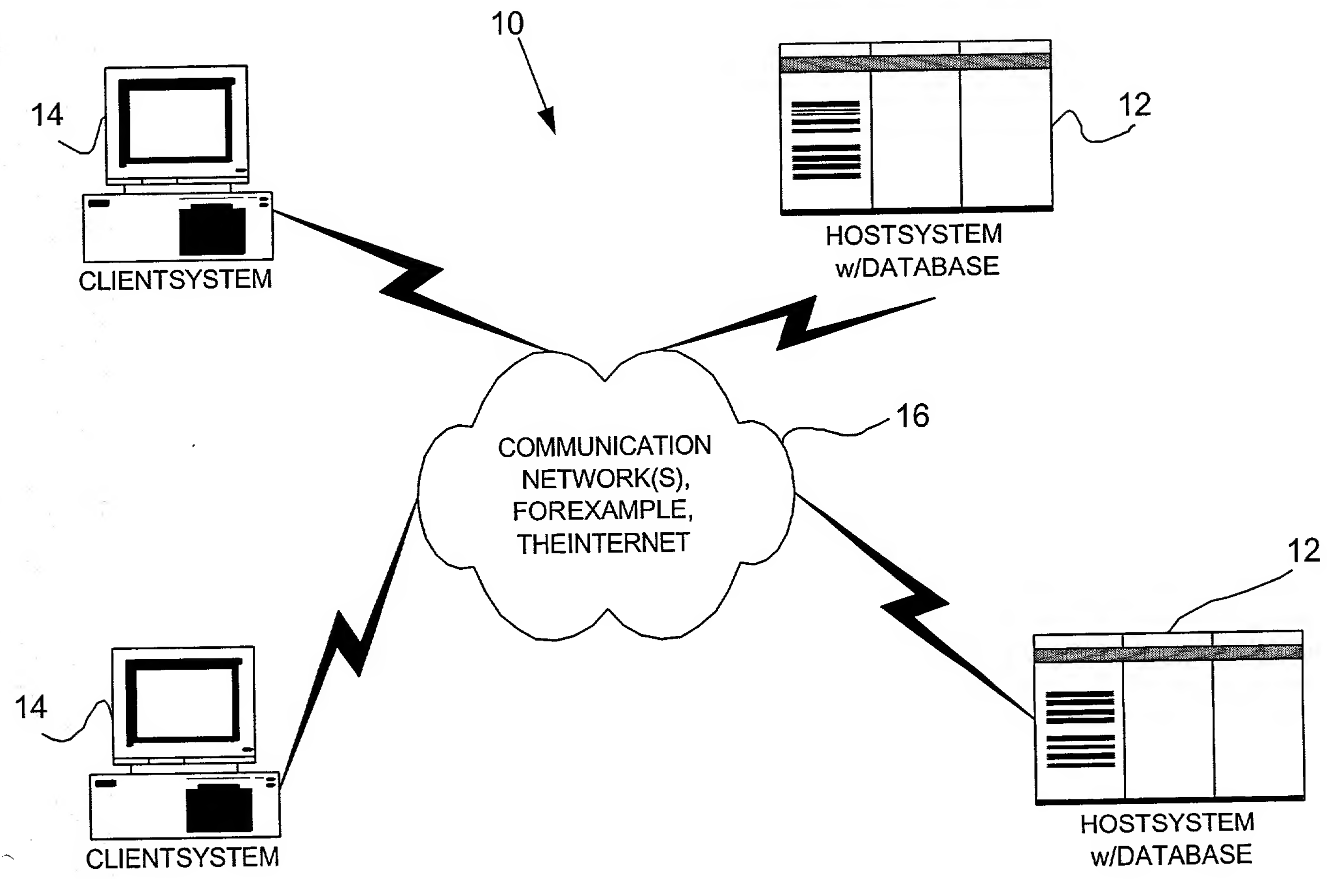

10
14
CLIENTSYSTEM
12
HOSTSYSTEM w/DATABASE
COMMUNICATION NETWORK(S), FOREXAMPLE, THEINTERNET
16
14
CLIENTSYSTEM
12
HOSTSYSTEM w/DATABASE

Fig.5

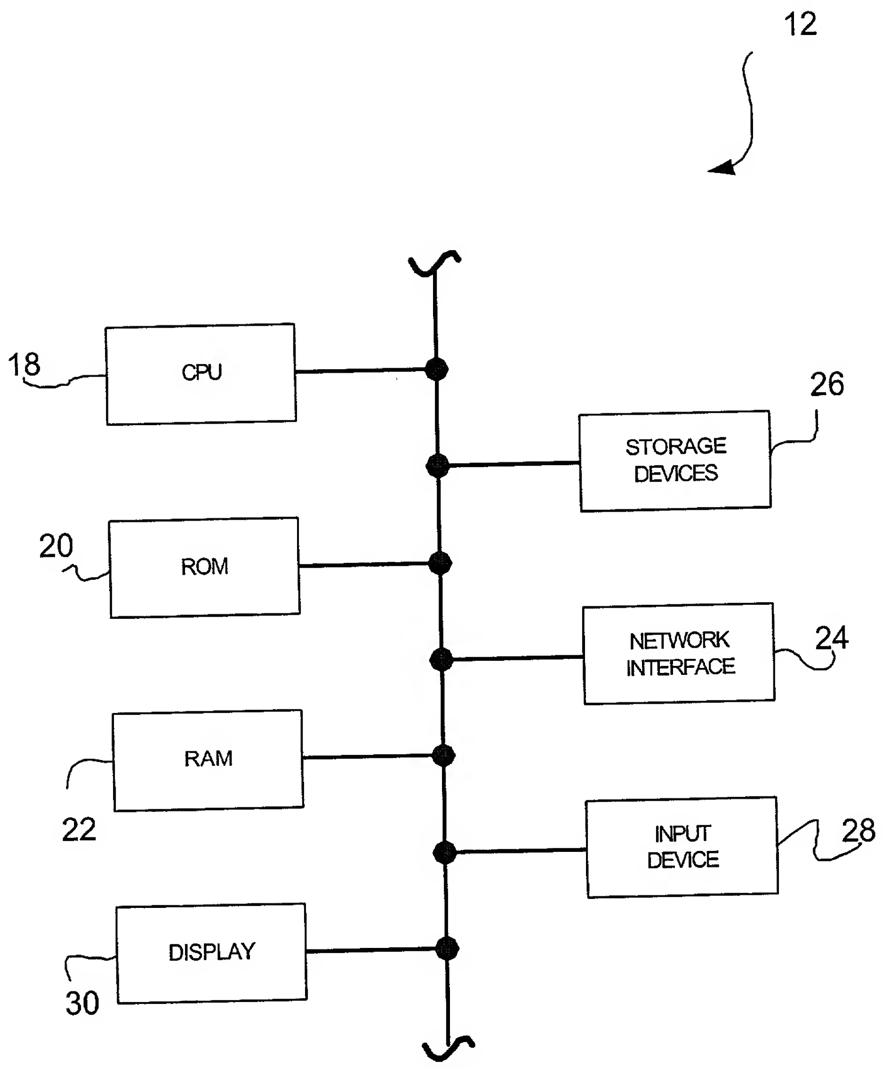

12
CPU
18
STORAGE DEVICES
26
ROM
20
NETWORK INTERFACE
24
RAM
22
INPUT DEVICE
28
DISPLAY
30

Fig.6

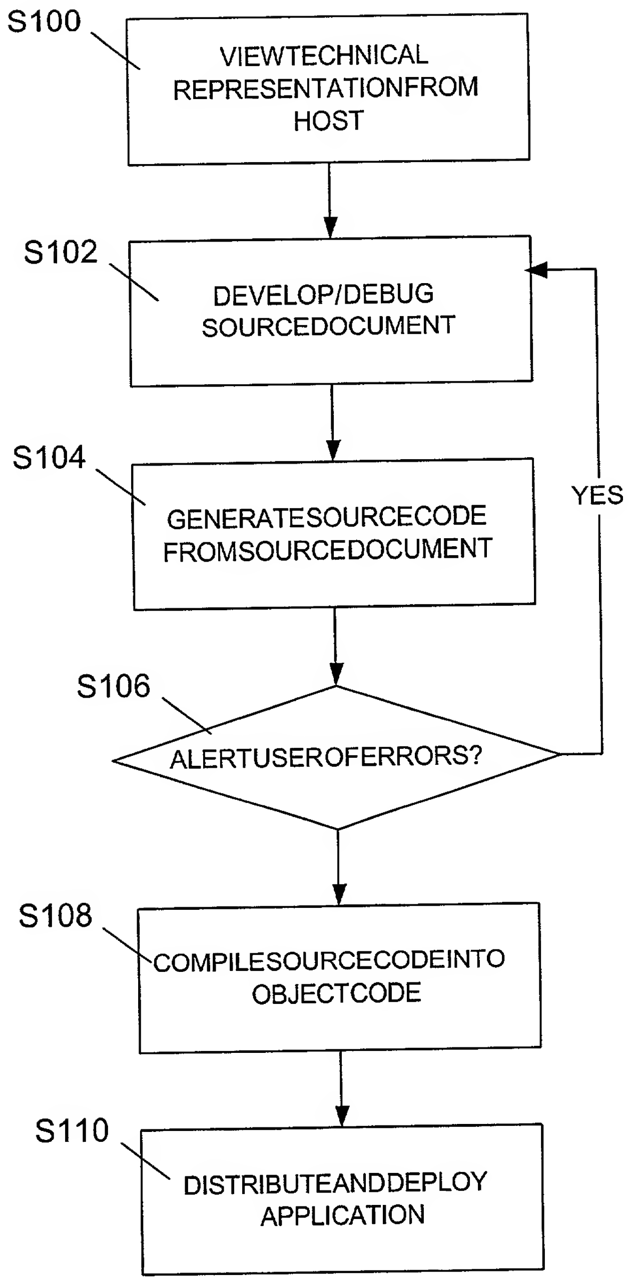

S100
VIEWTECHNICAL REPRESENTATIONFROM HOST
S102
DEVELOP/DEBUG SOURCEDOCUMENT
S104
GENERATESOURCECODE FROMSOURCEDOCUMENT
YES
S106
ALERTUSEROFERRORS?
S108
COMPILESOURCECODEINTO OBJECTCODE
S110
DISTRIBUTEANDDEPLOY APPLICATION

Fig.7

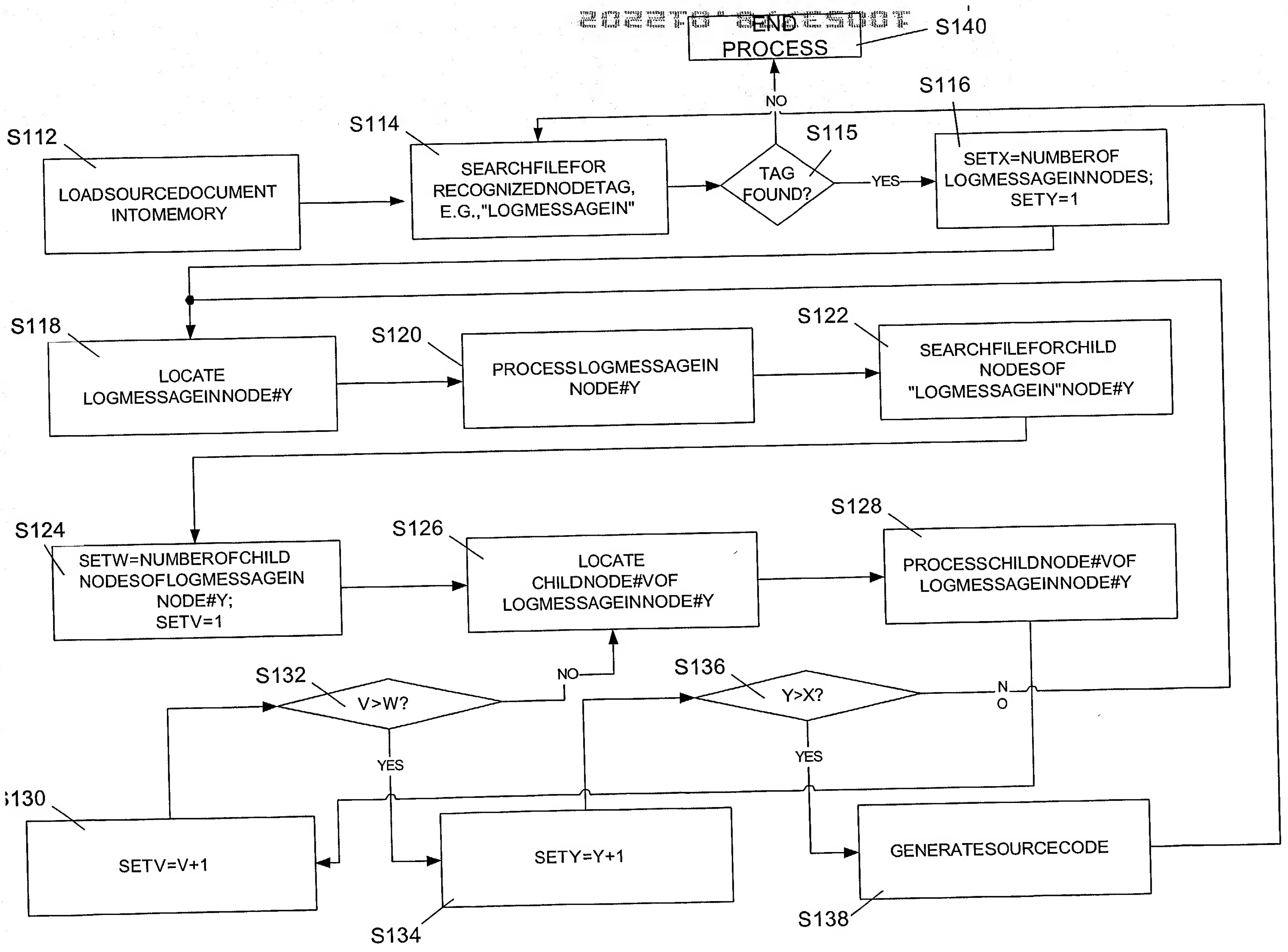

S140
END PROCESS
S116
NO
S112
S114
S115
LOADSOURCEDOCUMENT INTOMEMORY
SEARCHFILEFOR RECOGNIZEDNODETAG, E.G.,"LOGMESSAGEIN"
TAG FOUND?
YES
SETX=NUMBEROF LOGMESSAGEINNODES; SETY=1
S118
S120
S122
LOCATE LOGMESSAGEINNODE#Y
PROCESSLOGMESSAGEIN NODE#Y
SEARCHFILEFORCHILD NODESOF "LOGMESSAGEIN"NODE#Y
S124
S126
S128
SETW=NUMBEROFCHILD NODESOFLOGMESSAGEIN NODE#Y; SETV=1
LOCATE CHILDNODE#VOF LOGMESSAGEINNODE#Y
PROCESSCHILDNODE#VOF LOGMESSAGEINNODE#Y
S132
V>W?
NO
S136
Y>X?
NO
YES
YES
S130
SETV=V+1
SETY=Y+1
GENERATESOURCECODE
S134
S138

Fig. 8

```
if(!FAILED(spElement->getElementsByTagName(
        L"LogMessageIn",&spNodes))&&spNodes)
{
        longlLength=0;
        if(FAILED(spNodes->get_length(&lLength)))
        {
                return;
        }
        CStringsText;
        for(longi=0;i<lLength;       i++)
        {
                spNode.Release();
                if(!FAILED(spNodes->get_item(i,&spNode)))
                {
                        CheckOutputLine(0,spNode);
                        GetAttribute(spNode,L"id",sText);
                        if(m_bDoVB)
                        {
                sText=_T("SubLogMessageIn_")+sText+_T("(vtObjasVTMsgObj)");
                                OutputLine(0,sText);
                                OutputLine(0,_T("\r\n"));
                                OutputLine(1,_T("'Tempvariablesusedbyroutine\r\n"));
                                OutputLine(1,_T("DimsTmpAsString\r\n"));
                                OutputLine(1,_ T("DimsTmp4AsString\r\n"));
                                OutputLine(1,_T("DimsTmp5AsString\r\n"));
                                OutputLine(1,_T("DimsTmp3AsString\r\n"));
                                OutputLine(1,_T("DimsTmp2AsString\r\n"));
                                OutputLine(1,_T("DimsCmpAsString\r\n"));
                                OutputLine(1,_T("DimiOffsetAsinteger\r\n"));
                                OutputLine(1,_T("\r\n"));
                                OutputLine(1,_T("OnErrorGotoErrOut\r\n"));
                                OutputLine(1,_T("\r\n"));
                        }else
                        {
                                AddCFunction("LogMessageIn_"+sText    );
                                sText=_T("voidMsgHandler::LogMessageIn_")+sText;
                                sText+=_T("(IDualVTMsgObj*vtObj)\r\n{\r\n\ttry{\r\n");
                                OutputLine(0,sText);
                                OutputLine(0,_T("\r\n"));
                                OutputLine(1,_T("//Tempvariablesusedbyroutine\r\n"));
                                OutputLine(1,_T("CComBSTRsTmp;\r\n"));
                                OutputLine(1,_T("CComBSTRsTmp2;\r\n"));
                                OutputLine(1,_T("CComBSTRsTmp3;\r\n"));
                                OutputLine(1,_T("CComBSTRsTmp5;\n"));
                                OutputLine(1,_T("CComBSTRsTmp4;\n"));
                                OutputLine(1,_T("CComBSTRsCmp;\r\n"));
                                OutputLine(1,_T("intiOffset=0,iLastPos=0;\r\n"));
                                OutputLine(1,_T("\r\n"));
                                OutputLine(1,_T("//Endoftempvariables\r\n"));
                                OutputLine(1,_T("  \r\n"));
                        }

                        m_bProcessingGetResponseCode+=1;
                        ProcessMessageIn(1,spNode);
                        m_bProcessingGetResponseCode-=1;

                        if(m_bDoVB)
                        {
                                OutputLine(0,_T("ErrOut:\r\n"));
                        OutputLine(1,_T("vtObj.EndSetErrorErr.Number,Err.Description\r\n"));
                        }
```

Fig.9

```
voidCXMLEditDoc::ProcessMessageIn(intiTabIndex,CComPtr<IXMLDOMNode>&
spParentNode,BOOLbInBuildField,BOOL*bIfWasProccessed)
{
        CComPtr<IXMLDOMNode>spChild;
        if(FAILED(spParentNode->get_firstChild(&spChild))||!spChild)
        {
                return;
        }

        CComPtr<IXMLDOMNodeList>spList=NULL;
        if(!FAILED(spParentNode->get_childNodes(&spList))&&spList)
        {
                CComBSTRsNodeName;
                CComBSTRsText;
                CComPtr<IXMLDOMNode>spNode;
                longlLength=0;
                spList->get_length(&lLength);
                for(longi=0;i<lLength;i++)
                {
                        spNode.Release();
                        sText.Empty();
                        sNodeName.Empty();
                        if(!FAILED(spList->get_item(i,&spNode)))
                        {
                                spNode->get_nodeName(&sNodeName);

                                CheckOutputLine(iTabIndex,spNode);

                                void*ptr=NULL;

                                CStringstrNodeName=sNodeName;

                                strNodeName.MakeUpper();

                                //nowtheuglylookuptable
                                glb_MapOfXMLStringsToIds.Lookup(strNodeName,ptr);

                                intid=(int)   ptr;

                                switch(id)
                                {
*
*
                                caseIDTAG_BitmapDateIn:
                                {
                                        CStringsBitPos;
                                        CStringsLen;
                                        CStringsTranField;
*
*               {
                                                  OutputLine(iTabIndex,_T("If
vtObj.IsBitmapPositionSet("),_T("if(IsBitmapPositionSet(vtObj,"));

                                                  OutputLine(0,sBitPos);
                                                  OutputLine(0,_T(")Then\r\n"),_T(")){\r\n"));
                                                  iTabIndex+=1;
                                                  //firstletsgetthedata
                                                  CheckForPackedAttribute( iTabIndex, spNode );
```

Fig.10

10053728.042202

10053728.042202

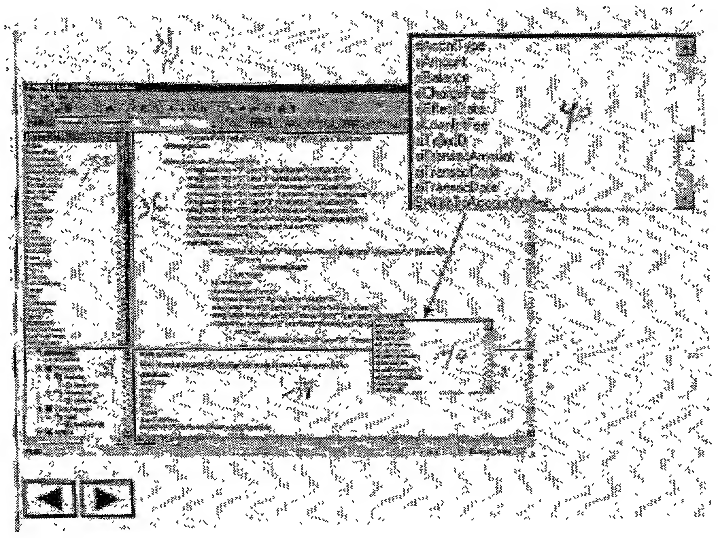


Fig.11